

1995122332

Geometry modeling and grid generation using  
3D NURBS control volume\*

Tzu-Yi Yu †  
Bharat K. Soni ‡  
Ming-Hsin Shih ★

Engineering Research Center for Computational Field Simulation  
Mississippi State University, MS 39762

### ABSTRACT

The algorithms for volume grid generation using NURBS geometric representation are presented. The parameterization algorithm is enhanced to yield a desired physical distribution on the curve, surface and volume. This approach bridges the gap between CAD surface / volume definition and surface / volume grid generation. Computational examples associated with practical configurations have shown the utilization of these algorithms.

### INTRODUCTION

Surface grid generation is the most labor intensive part of the overall complex three dimensional grid generation process. Also, a significant amount of effort is required in changing the resolutions (grid sizes) and / or the distribution of the grid while maintaining geometry fidelity. In the last few years, various researchers have concentrated on utilizing the Computer Aided Geometry Design (CAGD) techniques [Ref 2,13,14] to expedite the overall surface generation process.

There are many approaches for representing sculptured geometry, such as rational or non-rational Bezier, parametric form, rational or non-rational B-spines, ... etc. Among these representation, the Non-Uniform Rational B-Splines (NURBS) has been widely accepted among these researchers. NURBS has been widely utilized to represent and design geometry in the CAD/CAM and the graphics community due to its powerful features, such as the local control property, variation diminishing, convex hull and affine invariance [Ref 1,2]. Also the geometry tool kits, such as curve/surface interpolation, data reduction, degree elevation, knot insertion and splitting, are well-developed [Ref 2,3,4]. These properties have made NURBS representation very popular in recent developments in CAD/CAM and hence in grid generation. For example, in 1992, the scientists of NASA research centers formed a NASA IGES (Initial Graphic Exchange Specification) committee and defined a format named NINO (NASA IGES NURBS ONLY) standard to encourage the CFD community to follow this standard with the NURBS definition for the geometry communication between systems [Ref 5].

The development of the software based on NURBS representation package: CAGI (Computer Aided Grid Interface) was initiated by authors [Ref 15] under the sponsorship of NASA Marshall Space Flight Cen-

---

\* This work is sponsored by NASA Marshall Space Flight Center.

† Graduate Student, AIAA Member

‡ Professor of Aerospace Engineering, AIAA Member.

★ Post-Doctor Fellow, AIAA Member.

ter. The purpose of this paper is to present the progress realized in enhancing the NURBS based curve / surface grid generation techniques into a 3D volume grid generation technique. To this end, various options for generating 3D volume geometry-grid are discussed. A reparameterization scheme has been developed to achieve desired distribution in physical space. Computational examples for modeling practical configurations have been exercised using the volume options and the reparameterization scheme.

## NURBS FORMULATION

Detailed mathematical discussion on NURBS can be found in [Ref 2,3,4]. A brief definition of NURBS curve / surface / volume is presented as follows:

A NURBS Curve of order  $k$  is defined as :

$$C(t) = \frac{\sum_{i=0}^n W_i d_i N_i^k(t)}{\sum_{i=0}^n W_i N_i^k(t)} \quad (1)$$

where the  $d_i$   $i=0, \dots, n$  denotes the deBoor control polygon and the  $W_i$  are the weights associated with each control point.  $N_i^k(t)$  is the normalized B-spline basis function of order  $k$  and is defined over a knot vector  $T=T_i$   $i=0, \dots, n+k$  by the recurrence relations as shown in equation (2).

$$N_i^k(t) = \frac{(t - T_i) N_i^{k-1}(t)}{T_{i+k-1} - T_i} + \frac{(T_{i+k} - t) N_{i+1}^{k-1}(t)}{T_{i+k} - T_{i+1}} \quad (2)$$

$$N_i^k(t) \begin{cases} = 1 & \text{if } T_i \leq t < T_{i+1} \\ = 0 & \text{otherwise} \end{cases}$$

Throughout this paper, it is assumed that the knot vector has the form  $T = \{0, \dots, 0, T_k, \dots, T_n, 1, \dots, 1\}$  with the multiplicity  $k$  for the knot value  $0$  and  $1$  on both ends of the knot vector. If the knot vectors do not match this format, the knot insertion [Ref 6,7] technique must be used to achieve the multiplicity of  $k$  on the ends of the knot vector, and if the end knot values are not  $0$  and  $1$ , the knot vector must be normalized by the last knot value to match this format.

The NURBS surface is the extension of the curve from 1D to the 2D tensor product parametric surface and is shown as equation (3).

$$S(s, t) = \frac{\sum_{i=0}^m \sum_{j=0}^n W_{ij} d_{ij} N_i^{k1}(s) N_j^{k2}(t)}{\sum_{i=0}^m \sum_{j=0}^n W_{ij} N_i^{k1}(s) N_j^{k2}(t)} \quad (3)$$

Where the  $d_{ij}$  denotes the 3D control net and the  $W_{ij}$  are the weights associated with each control points.  $N_i^{k1}(s)$ ,  $N_j^{k2}(t)$  denote the normalized B-Spline basis functions of order  $k1$  and  $k2$  in the  $I$  and  $J$  directions, respectively.

The formula for 3D NURBS volume is defined analogous to NURBS surface and is a 3D tensor product form written as equation (4).

$$V(s,t,u) = \frac{\sum_{i=0}^m \sum_{j=0}^n \sum_{l=0}^p W_{ijl} d_{ijl} N_i^{k1}(s) N_j^{k2}(t) N_l^{k3}(u)}{\sum_{i=0}^m \sum_{j=0}^n \sum_{l=0}^p W_{ijl} N_i^{k1}(s) N_j^{k2}(t) N_l^{k3}(u)} \quad (4)$$

The  $d_{ijl}$  form the 3D control volume and the  $W_{ijk}$  are the weights associated with each control points. And  $N_i^{k1}(s)$ ,  $N_j^{k2}(t)$  and  $N_l^{k3}(u)$  are the normalized B-spline basis functions of order  $k1$ ,  $k2$  and  $k3$  in the  $I$ ,  $J$  and  $L$  directions (Instead of using  $I, J, K$ , this paper uses  $I, J, L$  to denote the three directions), respectively.

### 3D NURBS CONTROL SURFACE

NURBS has been used to model geometry in CAD/CAM for a long time. Manipulation tools such as “knot insertion” and “degree elevation” are used to increase the flexibility of manipulating the entire geometry. The famous algorithms for knot insertion are “Boehm” algorithm [Ref 6] and “Oslo” algorithm [Ref 7]. Both knot insertion and degree elevation are fundamental and powerful tools and are very frequently used in modeling the NURBS geometry. The curve and surface interpolation [Ref 8,11] techniques are also used to transform the discrete geometry to B-spline definition. These algorithms have been used as basic tools in modeling the NURBS geometry. Also, it has been shown [Ref 8,9] that many geometries can be represented analytically with a very concise control polygon (control net). For example, a NURBS circular arc can be defined with only 3 control points. Instead of storing the surface grid points, one can store the associated control polygon (control net or volume for the surface and the volume) with the associated weights to reduce the memory load, and this has been considered as one of the features of the NURBS representation.

The algorithms to construct NURBS representation for ruled surface, extruded surface, surface of revolution ...,etc. along with analytical geometries with appropriate data reduction algorithm are well developed and well documented in the CAGD literature. These algorithms have been enhanced and incorporated in the CAGI system [Ref 15]. The application of these algorithms to practical CFD related grid configurations is demonstrated in Figures 1~4. Figure 1 shows four NURBS control patches and their surfaces. This multiple-duct engine is created first by reading the curve profiles, interpolating these discrete data set to NURBS curves, then performing the data reduction [Ref 10] to reduce the redundant control points, and then finally, by using the NURBS revolution [Ref 3,4,13] algorithm. Similar procedures applied to create the single rotation propfan shown in Figure 4. The teapot shown in Figure 2 is modeled by three NURBS control nets, while the missile configuration with four fins shown in Figure 3 is generated by the eight control nets.

### 3D NURBS CONTROL VOLUME

The widely used technique to algebraically generate the three dimensional volume grid is by utilizing the transfinite interpolation algorithm based on the bounding surface grids. The ultimate objective of the present research effort is to explore various NURBS control volume options applicable to three dimensional grid generation. In this paper progress realized in the development of ruled NURBS volume, extruded volume, volume of revolution and composite volume are discussed.

#### Ruled NURBS volume.

The easiest way to form a 3D NURBS volume is the ruled NURBS volume. The algorithm is described as followed: Given two NURBS surfaces, the first step to form a ruled volume is making the knot vectors of the surfaces be in the same range of [0~1]. Next, considering the  $I$  direction of both surfaces, use the degree raising technique to raise the low degree of the surface. This procedure will yield a new knot vector and new control net. If the new knot vector differs from the other knot vector, then perform knot

insertion algorithm to merge them into one final knot vector. Then these entire procedures must be applied to the  $J$  direction of both surfaces again. After this step, the two NURBS surface will have the same orders and the same knot vectors in both  $I$  and  $J$  directions. This means the resolutions of control net of both surfaces will be the same. Therefore, one can connect the corresponding control point together to form the 3D NURBS volume. In other words, the orders and knot vectors of the final volume in  $I$  and  $J$  directions will be the same as those of the surfaces after degree elevation and knot insertion, and the order in  $L$  direction will be set as two with knot vector set as (0,0,1,1). Figure 5 shows a 3D “apple-like” NURBS volume formed by this algorithm.

While defining the NURBS ruled surface, the IGES defines a variable named “*DIRFLG*” as a direction flag to control the direction of how the surface will be connected. If *DIRFLG* is 0, then the surface will connect the points on the same direction of the two curves, otherwise, the direction of one of the curves will be reversed. Similar to this definition, it is possible to set two flags as “*DIRFLG\_I*” and “*DIRFLG\_J*” to control how the directions of the two surfaces will be connected. This increases the flexibility of generation options.

#### Extruded volume.

IGES defines the extruded surface as a surface formed by moving a line segment (called generatrix) parallel to itself along a curve(called directrix). In other words, given a NURBS curve, one can generate another curve by extruding the given curve with a distance  $\alpha$  along a vector  $V$ . Similar to this definition, the NURBS extruded volume is defined as: given a NURBS surface, a new surface can be generated by extruding the given surface with a distance  $\alpha$  along a vector  $V$ . Mathematically, this new extruded surface can be described as  $\vec{d}_{ij} = d_{ij} + \alpha \vec{V}$  with the same orders, same knot vectors as those of the given surface. After this step, the algorithm of “ruled volume” can be applied to these surfaces to form a final NURBS volume. Figure 6 shows a 3D NURBS extruded volume.

#### Volume of revolution.

IGES also defines the surface of revolution entity as the surface which is formed by rotating a given curve (called generatrix) with respect to an arbitrary straight line (axis of revolution) from a starting angle (not necessarily zero) to an ending angle. Compared to most of the literature which discussed the full revolution (rotation angle is set to  $360^\circ$ ), this definition is more general and useful in grid generation. The NURBS volume of revolution discussed in this paper is the extension of the surface of revolution, and it is formed by rotating a given NURBS surface with respect to an arbitrary line as the axis of revolution from a starting angle to an ending angle. This general algorithm can be stated as follows: First step is translating /rotating the axis of revolution by proper transformation matrix so that it is coincident with the Z axis. Also, apply this transformation matrix to the given NURBS surface so that the entire surface can be kept in the same position as the axis of rotation. It is assumed that the surface is defined as NURBS with the control net  $d_{ij}$ , order  $k1$  and  $k2$ , weights  $W_{ij}$  and two knot vectors. Next, for each control net  $d_{ij}$  (on the generatrix  $i=0,\dots,m, j=0,\dots,n$ ), construct the control volume  $d_{ijl}$   $l=0 \dots p$  at each  $j$ -th cross section according to the starting and ending angle by utilizing the circular arc algorithm. In other word, this approach constructs the NURBS control net at each  $j$  constant plane by revolving the control polygon  $d_{ij}$  with respect to  $L$  direction and then “stacks” them together to form a final NURBS volume. Figure 7 demonstrates this approach. The procedure for generating the NURBS circular arc can be referenced in [Ref 9,13] and the  $p$  (for the last dimension of control volume) is determined by the sector angle (equal to the difference between ending and starting angle). For example, if the angle is less than  $90^\circ$ ,  $p$  is equal to 2. If the angle is in the range of  $90^\circ \sim 180^\circ$ ,  $p$  is equal to 4, if in the range of  $180^\circ \sim 270^\circ$ ,  $p$  is 6, if it is greater than  $270^\circ$ ,  $p$  should be 8. For the section angle  $\theta$ , the weights are set as (in each  $J$  constant plane,  $J=0,\dots,n$ )  $W_{ijp} = w_{ij}, w_{ij} \cos(\theta/p), w_{ij}, w_{ij} \cos(\theta/p), \dots$   $i=0,\dots,m$  (repeat  $w_{ij}, w_{ij} \cos(\theta/p)$  with total  $p+1$  terms). The knot vectors in directions of  $I$  ( $s$ ) and  $J$  ( $t$ ) are the same as the ones of the given surface while the knot vector in direction  $L$  ( $u$ ) is determined according to the circular arc procedure. For example, when  $p = 2$ , the associated knot vector is set as (0,0,0,1,1,1), for the case of  $p = 4$ , the knot vector is set as (0, 0, 0, 1/2, 1/2, 1, 1, 1), for the case of  $p = 6$ , the knot vector is set as (0, 0, 0, 1/3, 1/3, 2/3, 2/3, 1,1,1) and the last case when  $p = 8$ , the knot vector is (0,

0, 0, .25, .25, .5, .5, .75, .75, 1,1,1). Also, set the orders in  $I$  and  $J$  be  $k_1$  and  $k_2$  (as the ones in original surface) while set 3 as the order in  $L$  direction. Since the NURBS has the translate and rotate invariant property, one can apply the inverse transformation matrix to the control volume (without altering the weights and knot vectors) back to the original coordinates. Figure 8 shows a 3D NURBS control volume developed by this approach.

Several complex circular pipes can be constructed by using the combination of the aforementioned approaches. Figure 9 (a) and (b) are formed by first building a NURBS surface, then extruding it with a desired distance to form a cylinder pipe, extracting the last cross sectional surface from this cylinder pipe and then performing the volume of revolution algorithm to create those turning portions, and again extracting the last cross sectional surface and using the ruled surface algorithm to form the last pipe. Last, the composite method should be performed to construct the entire volume pipe.

### Composite volume.

A composite NURBS volume is defined as a volume consisting of lists of constituent volumes. The compositing procedure is stated as follows: Suppose two constituent NURBS volume  $V_1$  and  $V_2$  form a composite volume. Assume that  $V_1$  has control volume  $d_1[0:m_1, 0:n_1, 0:l_1]$ , weight  $W_1[0:m_1, 0:n_1, 0:l_1]$ , three knot vectors  $knot\_i)_1, knot\_j)_1, knot\_l)_1$  and orders  $k\_i)_1, k\_j)_1, k\_l)_1$  while  $V_2$  has control volume  $d_2[0:m_2, 0:n_2, 0:l_2]$ , weight  $W_2[0:m_2, 0:n_2, 0:l_2]$ , three knot vectors  $knot\_i)_2, knot\_j)_2, knot\_l)_2$  and orders  $k\_i)_2, k\_j)_2, k\_l)_2$ . There are many combinations when two volumes are joined together. For example, one may join the volumes in  $I$  direction with the interface of  $J, L$  surface, or join in  $L$  direction with the interface of  $I, J$  surface, .. etc. Even though there are many cases, the procedure is similar. Take the case when joining in  $I$  direction as an example, the first step is to perform the degree elevation to  $V_1$  and  $V_2$  so that these two volumes can have the compatible degrees in  $I, J$  and  $L$  directions. If the two knot vectors in  $J$  direction for  $V_1$  and  $V_2$  are not the same, merging them together by setting the final knot vector as  $\{ knot\_j)_1 \cup knot\_j)_2 \}$ , then applying the knot insertion to  $V_1$  and  $V_2$  in  $J$  dimension. Same procedure should be applied to  $L$  direction if  $knot\_l)_1$  and  $knot\_l)_2$  are not the same. After this step,  $V_1$  and  $V_2$  will have the same degree in three directions, and the number of control points and knot vectors in  $J$  and  $L$  directions will be the same. The second step is to adjust the knot vector  $knot\_i)_2$  so that its first knot value can be the same as the last knot value  $knot\_i)_1$ . Shifting the knot vector will not change the original NURBS because the basis function is a "normalized" basis function. The third step is to build up the final knot vector by joining the two knot vectors into one knot vector and set that knot value at the joint point to have the multiplicity equal to  $(order - 1)$ . For example, if the knot vector  $knot\_i)_1$  is  $[0, 0, 0, 1, 1, 1]$  and the knot vector  $knot\_i)_2$  is  $[2, 2, 2, 3, 3, 3]$ , adjust the second knot vector by shifting -1 to each value. Thus, the  $knot\_i)_2$  becomes  $[1, 1, 1, 2, 2, 2]$ . Suppose the final  $order$  of these two volumes in  $I$  direction is three, then, the final knot vector should be  $[0, 0, 0, 1, 1, 2, 2, 2]$  (one may notice the interior knot 1 has multiplicity of  $(order-1)=2$ ). The fourth step is to match the weights. This step can be illustrated as the following pseudo code.

```

for (L=0; L<=l1; L++)
for (j=0; j<=n1; j++)
{
    let factor = W1[m1, j, L] / W2[0, j, L] ;

    for (i=0; i<=m2; i++)
        W2[i, j, L] = W2[i, j, L] *factor ;
}

```

The last step is to build up the final control volume and weights by throwing the  $d_2[0:0, 0:n_2, 0:l_2]$  and  $W_2[0:0, 0:n_2, 0:l_2]$  away and jointing the others as one control volume and weights.

## REPARAMETERIZATION PROCEDURE

As discussed earlier, it is clear that the NURBS representation is defined in a parametric form. Take a curve as an example: any value  $t$  in parametric space will result in a point  $C(t)$  in the physical space. This

property leads to some advantages and disadvantages. In perspective of numerical grid generation, the most conspicuous contributions for NURBS are that it provides a very easy and intuitive way for changing topologies, resolutions and the distributions of a surface grid. The surface grids generated in grid generation tools often have different topologies – such as O type, C type and H type grid or even unstructured or hybrid grid. Generating a 3D surface from these different topologies may require repeated applications. However, by using the 3D NURBS control polygon and weights the re-mapping and redistribution can be achieved by re-establishing the associated distribution mesh (network of parametric values). Figure 10~11 demonstrate the examples of the redistribution process. In these example, the structured and unstructured (hybrid) grid is generated just by constructing the associated hybrid distribution space. The other advantages are dealt with the resolution and distribution. Due to this parametric property, evaluating the same NURBS information (same control points, weights, knot vector and the same orders) with different parametric mesh (includes the different dimension of grid sizes) can yield a surface grid redistribution. This is accomplished by the “reparameterization” procedure.

What makes NURBS so popular in CAD design is that it provides many stable geometry manipulating properties. One of those properties is the local control property. The engineer can locally modify the NURBS geometry by changing the NURBS information (such as the location of the control polygon, weights or even knot vectors) without perturbing the entire geometry. However, this reshaping also changes the effect of parametric space on the distribution of points, often resulting in unacceptable distribution. This is due to “bad parameterization”— which means after changing any of the NURBS information (control polygon, knot vectors or weights), the desired distribution in parametric space will not necessarily result in the desired distribution in physical space (take equation (1) as an example, the evenly distribution in  $t$  (parametric space) doesn't result in evenly distribution in  $C(t)$  (physical space)). These situations are demonstrated in Figure 12. The curve itself has been locally modified, yet one can also notice that the points on the modified curve are stretched towards the control point for which the weight has been increased. This has been claimed as the disadvantage of the parametric NURBS. Several literatures [Ref 13,14] have shown their approaches to overcome this obstacle and to obtain the desired distribution in the final geometry after any modification. A new and more efficient algorithm for reparameterizing parametric curve has been developed as follows:

Consider a NURBS curve with resolution  $ni$ . Let

- (1)  $t_1(i)$ ,  $i=1, \dots, ni$  be the unknown distribution which is used to generate the desired curve in physical space;
- (2)  $t_2(i)$ ,  $i=1, \dots, ni$  be the normalized chord length of the curve with desired distribution; and
- (3)  $t_3(i)$ ,  $i=1, \dots, ni$  be the normalized chord length of the curve evaluated at parametric values  $t_1(i)$ ,  $i=1, \dots, ni$ .

Take Figure 12 as an example, if the designer would like to have the final curve as shown in Figure 12(c), then  $t_2(i)$  will be a 1D array which contains the distribution packing towards two ends, and  $t_1(i)$  are the parametric values which are to be determined such that the  $t_3(i)$ , the normalized chord length of final curve, would be the same as  $t_2(i)$  (or  $|t_2(i) - t_3(i)|$  be minimized for all  $i=1, \dots, ni$ ).

In order to obtain  $t_3(i)$ , the initial values of  $t_1(i)$  must be initialized. One can set  $t_1(i)$  as evenly distributed initially, then evaluate the curve with these parametric values to obtain the final curve, then normalize the entire chord length to obtain  $t_3(i)$ . Generally, the  $t_3(i)$  will be different than  $t_2(i)$ . How to adjust  $t_1(i)$  to have new  $t_3(i)$  which is close to  $t_2(i)$ ? The entire reparameterization procedure is described in the following computer pseudo code:

```

for (i=1; i<(ni-1); i++)
{
    find the location of  $j$  such that  $t_3[j] \leq t_2[i] < t_3[j+1]$ .

    let  $\alpha=t_2[i]-t_3[j]$ ;  $\beta=t_3[j+1]-t_2[i]$ ;
     $t[i]=(t_1[j+1]*\alpha + t_1[j]*\beta) / (\alpha + \beta)$ ;
}

```

The final  $t[i]$  will be the desired distribution curve, and using this distribution to evaluate the NURBS curve shown in Figure 12(b) (with the middle weight increased) will yield the result shown in Figure 12(c), which has the distribution packing towards two ends.

This reparameterization algorithm can also be extended to find the desired distribution mesh for the tensor product NURBS surface. The reasons which lead to the bad parameterization problems for the surface may come from arbitrarily modifying the NURBS information (such as the control point, weights or knot vector), or from significant difference of the arc lengths of two opposite boundaries. These cases are demonstrated in Figure 13 and 14. From equation (3), one can understand what cause the problem shown in Figure 14. There are only two knot vectors which control the distribution in  $I$  and  $J$  directions. If both of the two opposite boundaries in  $I$  (or  $J$ ) have different distributed requirements or have big difference in arc length, the one single knot vector in  $I$  (or  $J$ ) may cause problem. Similar to the reparameterization procedure of the curve, the algorithm for the surface is stated as followed.

Instead of defining the three 1D arrays, three sets of 2D arrays  $(s_1, t_1)$ ,  $(s_2, t_2)$  and  $(s_3, t_3)$  must be defined. Consider a NURBS surface with resolution  $n_i$  by  $n_j$ . Let

(1)  $(s_1(i,j), t_1(i,j))$ ,  $i=1, \dots, n_i, j=1, \dots, n_j$  be the unknown distribution mesh which is used to generate the desired distribution of the surface in physical space;

(2)  $(s_2(i,j), t_2(i,j))$ ,  $i=1, \dots, n_i, j=1, \dots, n_j$  be the normalized chord length of the surface with desired distribution; and

(3)  $(s_3(i,j), t_3(i,j))$ ,  $i=1, \dots, n_i, j=1, \dots, n_j$  be the normalized chord length of the surface evaluated at parametric values  $(s_1(i,j), t_1(i,j))$ .

Take Figure 13 as an example. If the designer would like to have the final surface, as shown in Figure 13(c), then  $(s_2(i,j), t_2(i,j))$  would be a 2D array which contains the even distribution, and  $(s_1(i,j), t_1(i,j))$  would be the parametric values which are to be determined such that the  $(s_3(i,j), t_3(i,j))$ , the normalized chord length of final surface, would be the same as  $(s_2(i,j), t_2(i,j))$  or within certain tolerance.

Similar to curve algorithm, one can set  $(s_1(i,j), t_1(i,j))$  as evenly distributed initially, then evaluate the surface with these parametric values to obtain  $(s_3(i,j), t_3(i,j))$ . Most likely the  $(s_3(i,j), t_3(i,j))$  will be different than  $(s_2(i,j), t_2(i,j))$ . The procedure to adjust  $(s_1(i,j), t_1(i,j))$  so that  $(s_3(i,j), t_3(i,j))$  can be close to  $(s_2(i,j), t_2(i,j))$  is presented with the following computer pseudo code:

```

for (j=1; j<(nj-1); j++)
for (i=1; i<(ni-1); i++)
{
  search the index of  $I, J$  such that  $(s_2(i,j), t_2(i,j))$  is located within the cell of  $(s_3(I, J), t_3(I, J))$ ,  $(s_3(I+1, J), t_3(I+1, J))$ ,  $(s_3(I, J+1), t_3(I, J+1))$ , and  $(s_3(I+1, J+1), t_3(I+1, J+1))$ ;

  let  $(s_2(i,j), t_2(i,j)) = (1-\alpha)(1-\beta)(s_3(I, J), t_3(I, J)) + (1-\alpha)\beta(s_3(I, J+1), t_3(I, J+1)) + \alpha(1-\beta)(s_3(I+1, J), t_3(I+1, J)) + \alpha\beta(s_3(I+1, J+1), t_3(I+1, J+1))$  and solve for  $\alpha$  and  $\beta$ ;

  new  $(s(i,j), t(i,j)) = (1-\alpha)(1-\beta)(s_1(I, J), t_1(I, J)) + (1-\alpha)\beta(s_1(I, J+1), t_1(I, J+1)) + \alpha(1-\beta)(s_1(I+1, J), t_1(I+1, J)) + \alpha\beta(s_1(I+1, J+1), t_1(I+1, J+1))$ ;
}

```

The final  $(s(i,j), t(i,j))$  are then the desired distribution mesh in which to generate the desired surface grid shown in Figure 13(c) and 14(d).

The reparameterization algorithm for the volume case is similar. 3 sets of 3D arrays should be used, and instead of bi-linear interpolation, the trilinear interpolation should be used while adjusting the desired distribution.

## CONCLUSIONS

The grid generation algorithm associated with the ruled volume, extruded volume, volume of revolution and the composite volume based on the NURBS representation have been presented and demonstrated

by utilizing practical configurations. The development of the reparameterization scheme and its influence to curve / surface / volume grid distribution is demonstrated by examples. These algorithms have been incorporated as modules in the CAGI system. The research associated with enhancing the volume options and manipulations along with evaluation of the traditional transfinite interpolation algorithm based on NURBS is underway.

## ACKNOWLEDGEMENT

The authors would like give their thanks to Mr. Ted Benjamin and Mr. Robert Williams of NASA Marshall research center for their support of this research project.

## REFERENCE

- [1] De Boor, C., - *A Practical Guide to Splines, Applied Mathematical Sciences, Vol 27, Springer-Verlag, 1972.*
- [2] FARIN, G. "Curves and Surfaces for Computer Aided Geometric Design: A Practical Guide," Third Edition, Academic Press, 1990.
- [3] Piegl, L., - 'On NURBS: A survey' *IEEE Computer Graphics & Applications*, Vol 11, No 1, pp 57 - 71, January, 1991.
- [4] Piegl, L. - 'Rational B-Spline Curves and Surfaces for CAD and Graphics' *State of the Art in Computer Graphics, Visualization and Modeling*, Eds, David F. Rogers, Rae A. Earnshaw, Springer-Verlag, pp 225-269, 1991.
- [5] Blake M.W., Kerr P.A., Thorp S.A., and Chou J.J., "NASA Geometry Data Exchange Specification for Computational Fluid Dynamics (NASA IGES)". NASA Reference Publication 1338., April, 1994.
- [6] Boehm, W. - 'Inserting New Knots into B-Spline Curves' *Computer Aided Design*, Vol. 12, No 4, pp 199~201, July 1980.
- [7] Cohen, E., Lyche, T. and Riesenfeld, R.F., - 'Discrete B-Splines and subdivision techniques in computer-aided geometric design and computer graphics', *Comput. Graph. & Image Proc.* Vol. 14, No 2, pp 87~111, 1980.
- [8] Tiller, W. - 'Rational B-Splines for Curve and Surface Representation' *IEEE Computer Graphics & Applications*, Vol. 3, No 10, pp 61~69, Sep 1983.
- [9] Piegl, L. and Tiller, W.- *A Menagerie of Rational B-Splines' IEEE Computer Graphics & Applications*, Vol 9, No 5, Sep 1989, pp 48~56.
- [10] Tiller, W.- 'Knot Removal Algorithms for NURBS Curves and Surface' *Computer Aided Design*, Vol 24, No 8, pp 445~453. August 1992.
- [11] Piegl, L. and Tiller, W.- 'Curve and surface constructions using rational B-splines' *Computer Aided Design*, Vol 19, No 9, 1987, pp 485~498.
- [12] Boehm W., Farin G. and Kahmann J.- 'A survey of curve and surface methods in CAGD' *Computer Aided Geometric Design*, pp 1~60, 1984.
- [13] Yu, T.Y. and Soni, B.K., "Geometry Transformer and NURBS in grid Generation," 4th International Conference on Numerical Grid Generation in CFD and Related Fields, Swansea, UK., April 6-8, 1994.
- [14] FARIN, G. "NURB curves and surfaces: from projective geometry to practical use," First Edition, A K Peters, Ltd., 1995.
- [15] Yu, T.Y., Soni, B.K., and Shih M. H. "CAGI: Computer Aided Grid Interface," AIAA-95-0243, 33rd Aerospace Sciences Meeting & Exhibit., Reno, Nevada January 09~12, 1995.

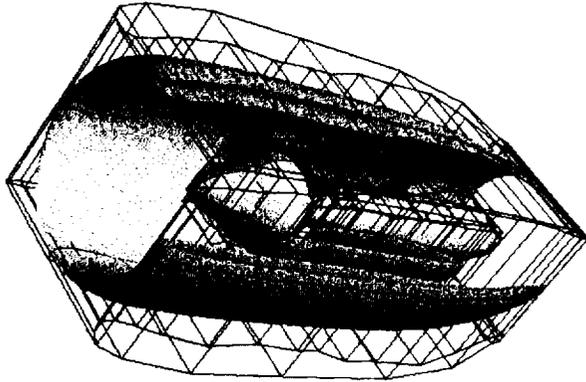


Figure 1. 3D NURBS control nets model the multiple-duct engine.

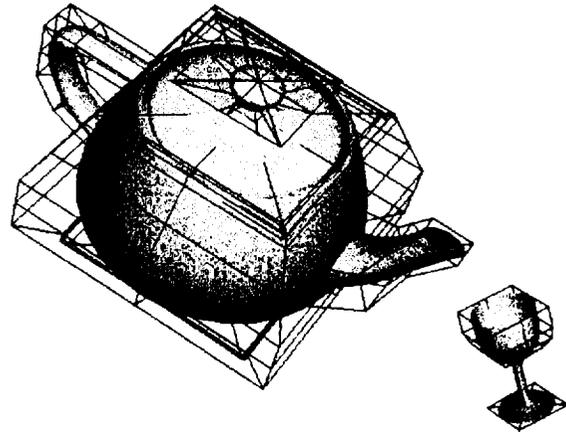


Figure 2. A teaset modeled by 3D NURBS control surfaces.

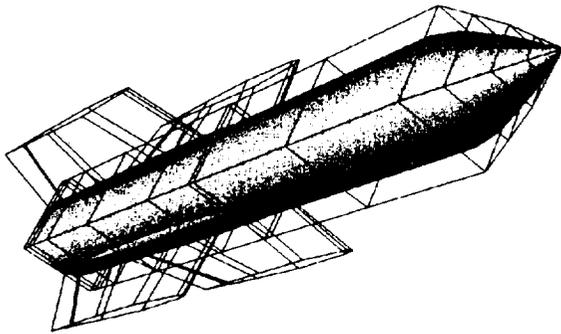


Figure 3. 3D NURBS control patches model the missile (with fins) geometry.

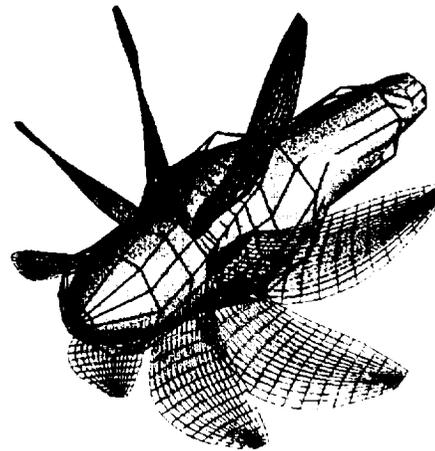


Figure 4. 3D NURBS control patches model the single rotation propfan.

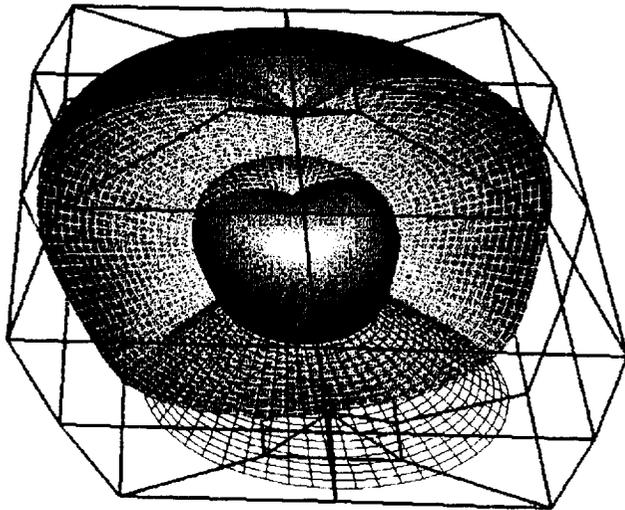


Figure 5. 3D NURBS control volume and its volume grid for ruled volume case.

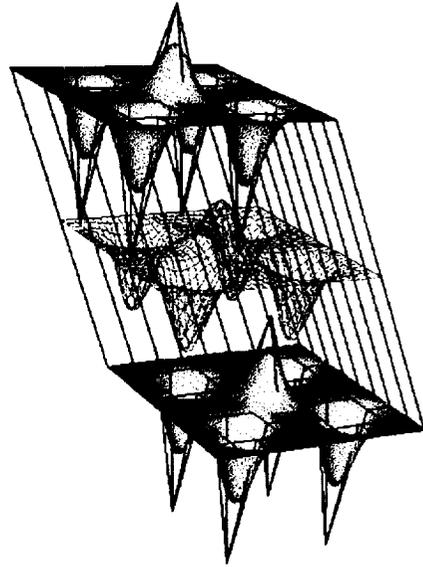


Figure 6. 3D NURBS extruded volume and its volume grid.

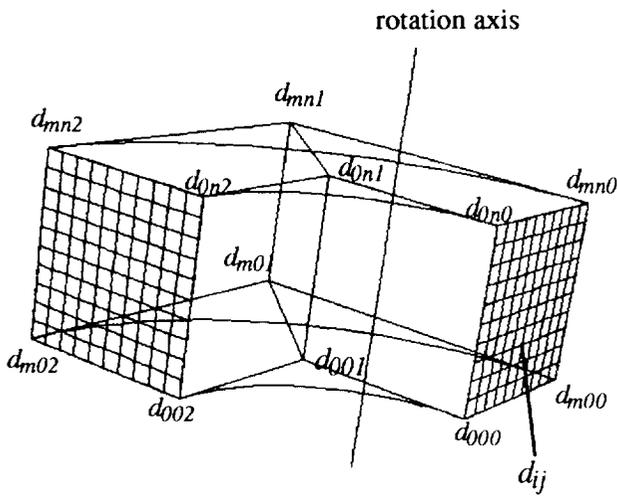


Figure 7. Illustration of volume of revolution.

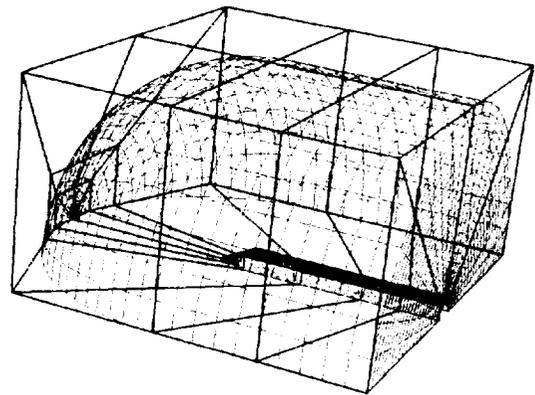


Figure 8. 3D NURBS control volume models the missile configuration.

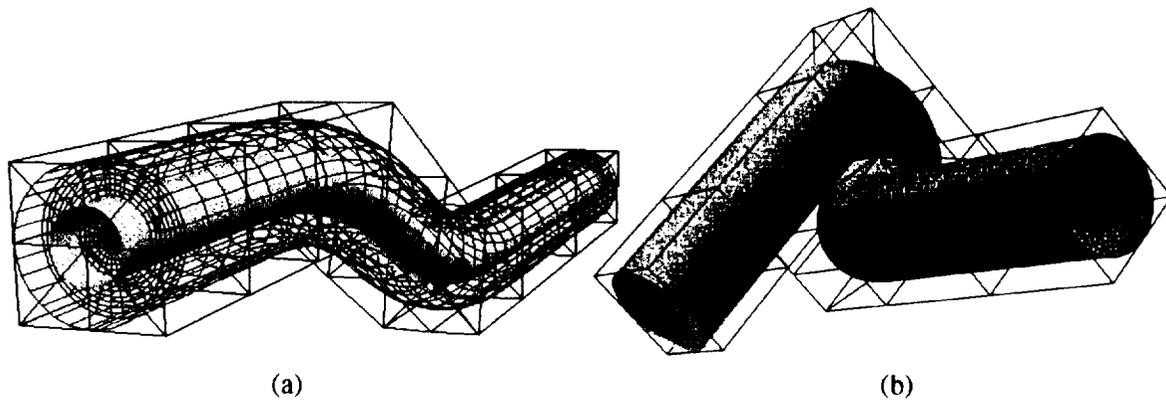
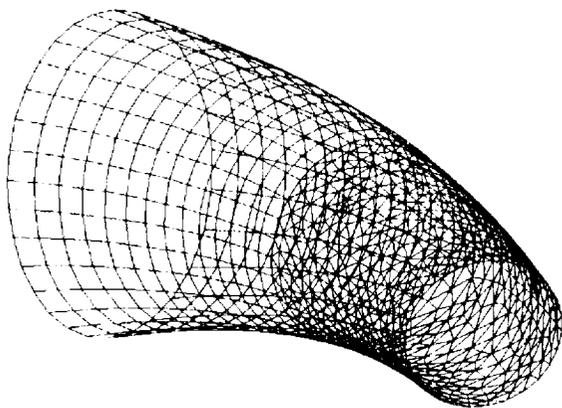
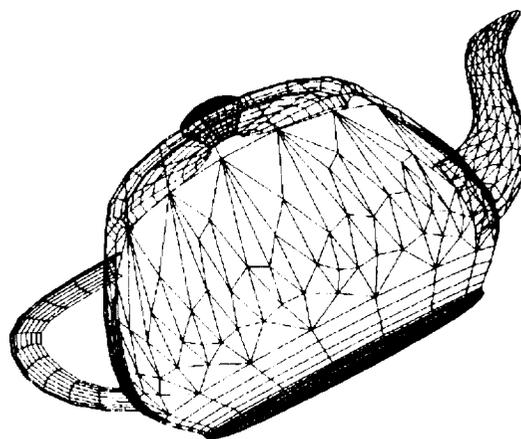


Figure 9 (a) (b). 3D NURBS circular pipes formed by the algorithm of ruled volume, extruded volume and volume of revolution.



(10)

Figure 10. 3D hybrid pipe.



(11)

Figure 11. 3D hybrid teapot.

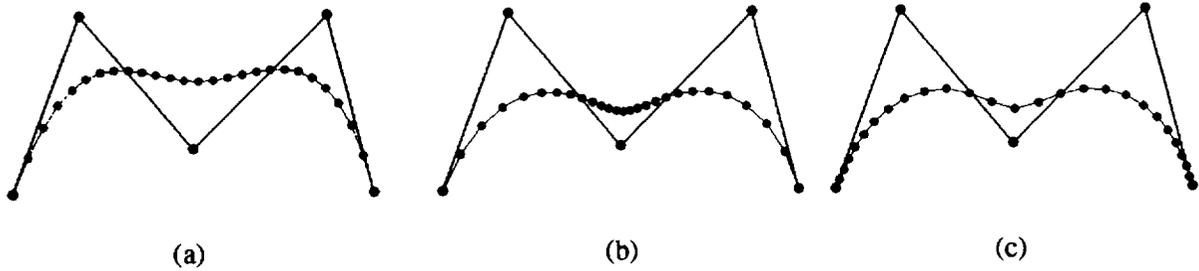


Figure 12. (a) Original NURBS curve and its control polygon.

(b) Increase the weight in the middle control point, evaluated the curve with even distribution in parametric space.

(c) Reparameterization example. The shape of the curve is unchanged, yet with new distribution packing towards two ends.

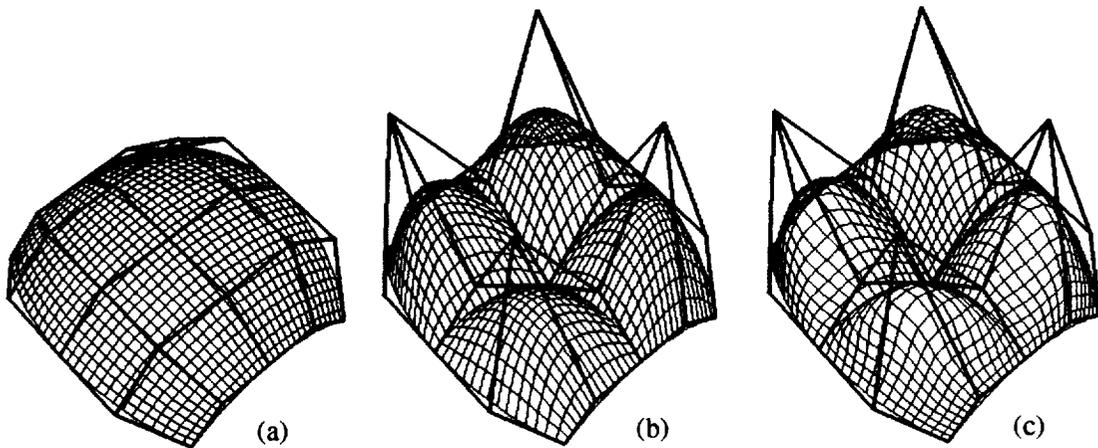


Figure 13. (a) Original NURBS surface and its control net.

(b) Alter the weights and control points location and evaluated the surface with even distribution in parametric space.

(c) Reparameterization example. The shape of the surface is unchanged, yet with new distribution.

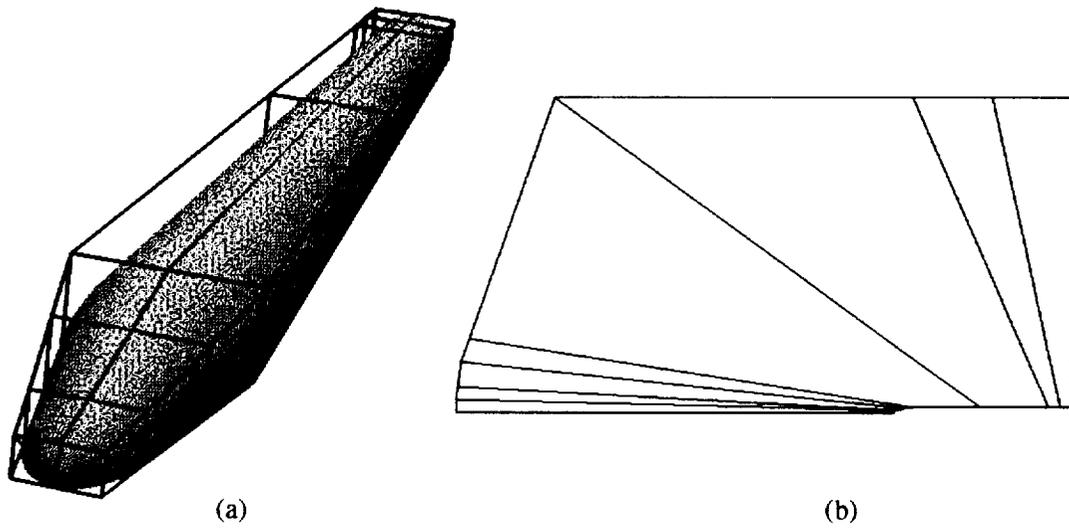


Figure 14 (a). A 3D control net models the missile configuration.  
(b). 2D missile control patch for external flow simulation.

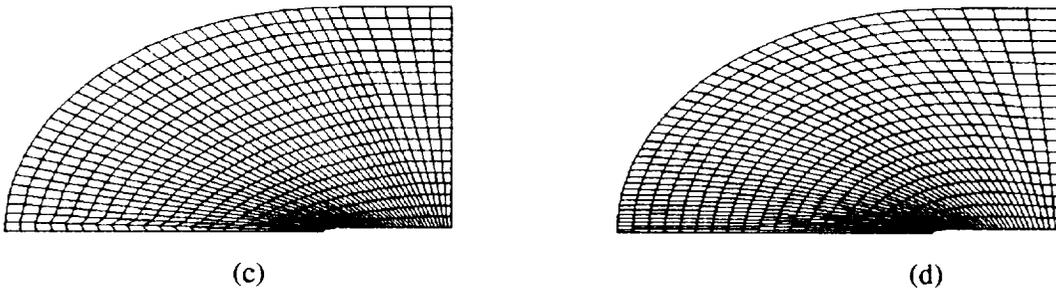


Figure 14 (c). 2D surface grid patch evaluated with even distribution in parametric space.  
(d). Reparameterization procedure for a smooth grid.

